降順に展開した漸化式を使用しないGBiCGSafe法の考案 (科学技術計算における理論と応用の新展開)

AUTHOR(S):

藤野, 清次

CITATION:

藤野, 清次. 降順に展開した漸化式を使用しないGBiCGSafe法の考案 (科学技術計算における理論と応用の新展開). 数理解析研究所講究録 2012, 1791: 31-36

ISSUE DATE:

2012-04

URL:

http://hdl.handle.net/2433/172842

数理解析研究所講究録
第1791巻 2012年 31-36

降順に展開した漸化式を使用しないGBiCGSafe法の考案

A proposal of GBiCGSafe method without reverse-ordered recurrence

藤野清次 (九州大学情報基盤研究開発センター)
Seiji FUJINO (Research Institute for Information Technology, Kyushu University)

Abstract:
We consider to solve a linear system of equations $Ax = b$ by iterative method. Product-type of iterative methods such as GPBiCG, BiCGSafe and GPBiCG_AR method constitute a polynomial list by multiplying Lanczos polynomials by acceleration polynomial. In this paper, we focus on the order of the development of recurrence for GPBiCG method and propose a GPBiCGSafe method by improving GPBiCG_AR method. Through numerical experiments, we make clear that the GPBiCGSafe method has excellent convergence rate and robustness.

1 はじめに

自然科学や工学などの様々な解析分野において，数値シミュレーションは重要な役割を果たしてきた．それらの様々な解析分野から生じる問題は，偏微分方程式で記述されることが多い．偏微分方程式は有限要素法などにより離散化することで，連立一次方程式に帰着される．この連立一次方程式の解法として反復法が提案されている．GPBiCG法やBiCGSafe法，GPBiCG_AR法などの積型反復法は，通常のランチョス多項式に加速多項式を掛けて，積の形の多項式列を構成することで反復法の収束を加速させるという発想に基づく反復法である．最近，阿部らによりGPBiCG法の変形版が提案された．そこでは，残差ベクトルの更新で用いられる加速多項式に交代漸化式の代わりに三項漸化式が適用された．以下では、それらをAS_GPBiCG_v1，AS_GPBiCG_v2法と呼ぶ．

本論文では，GPBiCG法において，交代漸化式から三項漸化式に変形するときに用いられる降順な漸化式の存在に着目し，それらを消すことによってGPBiCG_AR法の収束性の改良を図ったので，その結果を検証する．

本論文の構成は以下の通りである．第2節で，GPBiCG法の概要と漸化式の展開について述べ，GPBiCG_AR法を改良したGPBiCGSafe法を導く．第3節で，数値実験により，GPBiCGSafe法の収束性を検証する．第5節で，まとめと今後の課題を述べる．

2 積型反復法

2.1 GPBiCG法の算法の中の降順展開した漸化式

解くべき連立一次方程式を

$$Ax = b \tag{2.1}$$

とする．ただし，係数行列Aは大きさ$n \times n$の実非対称行列，xとbは次数nの解ベクトルと右辺ベクトルと各々する．初期近似解をx_0とし，初期残差ベクトルを$r_0 = b - Ax_0$とする．このとき，BiCG(Bi-Conjugate Gradient)法において，反復n回目の残差ベクトルr_n^{BiCG}と補助ベクトルp_n^{BiCG}は以下のように表される．

$$r_n^{\mathrm{BiCG}} = R_n(A)r_0,\ p_n^{\mathrm{BiCG}} = P_n(A)r_0. \tag{2.2}$$

多項式 $R_n(A)$ と $P_n(A)$ は次の交代漸化式を満たす.

$$\begin{aligned} R_0(\lambda) &= 1, P_0(\lambda) = 1, && (2.3)\\ R_n(\lambda) &= R_{n-1}(\lambda) - \alpha_{n-1}\lambda P_{n-1}(\lambda), && (2.4)\\ P_n(\lambda) &= R_n(\lambda) + \beta_{n-1}P_{n-1}(\lambda),\ n = 1, 2, \ldots. && (2.5) \end{aligned}$$

BiCG 法の残差列を $\{\boldsymbol{r}_0^{\mathrm{BiCG}}, \boldsymbol{r}_1^{\mathrm{BiCG}}, \ldots, \boldsymbol{r}_n^{\mathrm{BiCG}}\}$ とする. このとき, 適当な手続きにより多項式列 $\{H_0, H_1, \ldots, H_n\}$ を生成し, $\{H_0(A)\boldsymbol{r}_0^{\mathrm{BiCG}}, H_1(A)\boldsymbol{r}_1^{\mathrm{BiCG}}, \ldots, H_n(A)\boldsymbol{r}_n^{\mathrm{BiCG}}\}$ を構成することで, BiCG 法の残差列の収束の加速を図る. このように, 残差が BiCG 法の残差と加速多項式との積で定義された解法は積型反復法と呼ばれる. すなわち, 積型反復法の残差ベクトル $\boldsymbol{r}_n$ は

$$\boldsymbol{r}_n = H_n(A)\boldsymbol{r}_n^{\mathrm{BiCG}} = H_n(A)R_n(A)\boldsymbol{r}_0 \qquad (2.6)$$

と表される. 次に, 固有値 λ を使って多項式列 $\{G_n(\lambda)\}$ と $\{H_n(\lambda)\}$ を次の (交代) 漸化式で構成する.

$$\begin{aligned} H_0(\lambda) &= 1,\ G_0(\lambda) = \zeta_0, && (2.7)\\ H_n(\lambda) &= H_{n-1}(\lambda) - \lambda G_{n-1}(\lambda), && (2.8)\\ G_n(\lambda) &= \zeta_n H_n(\lambda) + \eta_n G_{n-1}(\lambda),\ n = 1, 2, \ldots. && (2.9) \end{aligned}$$

通常, 式 (2.8) と式 (2.9) の漸化式 H_n と G_n は, $H_1 \to G_1 \to H_2 \to G_2 \to H_3 \to G_3 \to \cdots$ の順番で更新される. すなわち, 式 (2.8) と式 (2.9) は昇順に展開される. 一方, GPBiCG 法では, 多項式列 $\{G_n(\lambda)\}$ は昇順に展開するのではなく, 次のように降順に展開して使用された. すなわち, n ステップ目の $G_n(\lambda)$ の多項式を次の $(n+1)$ ステップ目の $H_{n+1}(\lambda)$ を使って降順に展開した漸化式を使用した.

$$G_n(\lambda) = -(H_{n+1}(\lambda) - H_n(\lambda))/\lambda \qquad (2.10)$$

このように降順に展開した漸化式を使うと, 多項式列 $\{H_n(A)\}$ は次の 3 項から成る漸化式が得られるという利点がある. 一方, 数学的には同値でも, 降順展開された漸化式は, まるめ誤差の影響を受け易くなる可能性がある.

$$\begin{aligned} H_0(\lambda) &= 1,\ H_1(\lambda) = (1 - \zeta_0\lambda)H_0(\lambda), && (2.11)\\ H_{n+1}(\lambda) &= (1 + \eta_n - \zeta_n\lambda)H_n(\lambda) - \eta_n H_{n-1}(\lambda), n = 1, 2, \ldots && (2.12) \end{aligned}$$

ここで, ζ_n, η_n は新規のパラメータである. 生成された多項式列 $\{H_n(\lambda)\}$ は, 任意の n に対して $H_n(0) = 1$ を満たすので, $H_{n+1}(0) - H_n(0) = 0$ である. ここで, 漸化式 (2.12) は式 (2.9) を式 (2.8) に代入し, それを式 (2.10) を使って書き直すと容易に得られる. 以下では, 多項式 $H_n(\lambda), G_n(\lambda)$ を H_n, G_n と略記す.

ここでは, 算法を構成する過程で降順な漸化式 (2.10) が使われる様子を, 中間ベクトル $\boldsymbol{y}_n$ と $\boldsymbol{u}_n$ を取り上げ細かく観察する. すなわち, それらのベクトルの定義式と式変形は次のように各々書き表せる. 下線部分が降順の漸化式 (2.10) の部分である.

$$\begin{aligned} \underline{\lambda G_n} R_{n+2} &= \underline{(H_n - H_{n+1})} R_{n+2}\\ &= H_n(R_{n+1} - \alpha_{n+1}P_{n+1}) - H_{n+1}(R_{n+1} - \alpha_{n+1}P_{n+1}) && (2.13)\\ \lambda G_n P_n &= \lambda P_n(\zeta_n H_n + \eta_n G_{n-1})\\ &= \zeta_n\lambda H_n P_n + \eta_n(\underline{\lambda G_{n-1}} R_n + \beta_{n-1}\lambda G_{n-1}P_{n-1})\\ &= \zeta_n\lambda H_n P_n + \eta_n(\underline{(H_{n-1} - H_n)} R_n + \beta_{n-1}\lambda G_{n-1}P_{n-1}) && (2.14) \end{aligned}$$

2.2 GPBiCG_AR法

GPBiCG法の算法中の式(23)において，降順な漸化式を使わないようにした算法がGPBiCG_AR法である．前小節で考察したように，降順な漸化式を使わないようにするためには，中間ベクトル$\boldsymbol{y}_n$を使用しなければよいが，パラメターζ_nとη_nを決めることができなくなる．そこで，GPBiCG法のように残差ベクトルの2ノルムの最小化からではなく，准残差ベクトルの2ノルムの最小化からパラメータの値を定めた算法がGPBiCG_AR法である．この施策により，中間ベクトル$\boldsymbol{y}_n$(および$\boldsymbol{w}_n$)を使う必要がなくなったが，中間ベクトル$\boldsymbol{u}_n$はそのまま残ることになった．

2.3 GPBiCGSafe法

前述のように，GPBiCG_AR法において，中間ベクトル$\boldsymbol{u}_n$の更新において降順な漸化式が1度だけ使用されている．そこで，ベクトル$\boldsymbol{u}_n$をベクトル$\boldsymbol{z}_n$を用いて次のように書き表す．この式変形(下線を付けた部分)により降順な漸化式(2.10)の使用が消滅した．

$$\begin{aligned} \lambda G_n P_n &= \lambda P_n(\zeta_n H_n + \eta_n G_{n-1}) \\ &= \zeta_n \lambda H_n P_n + \eta_n(\underline{\lambda G_{n-1} R_n} + \beta_{n-1}\lambda G_{n-1} P_{n-1}) \end{aligned} \tag{2.15}$$

このように，降順な漸化式(2.10)を使わないよう改良した算法を**GPBiCGSafe**法と呼ぶ．また，下線部分に該当するベクトルの計算($A\boldsymbol{z}_{n-1}$)は従来のGPBiCG_AR法でも使用されていたため計算量は増えない．

3 数値実験

3.1 計算機環境と計算条件

計算はすべて倍精度浮動小数点演算で行った．計算機はNehalem（CPU：Intel Xenon X5570, クロック周波数：2.93GHz，メモリ：24Gbytes，OS：RedHat Enterprise Linux 5.6）を用いた．プログラムはFortran90を用いて実装し，最適化オプションは"-O3"を使用した．右辺項ベクトル$\boldsymbol{b}$は厳密解を$\hat{\boldsymbol{x}} = (1, 1, ..., 1)^T$とし，$\boldsymbol{b} = A\hat{\boldsymbol{x}}$で作成した．収束判定条件は相対残差の2ノルム：$||\boldsymbol{r}_{k+1}||_2/||\boldsymbol{r}_0||_2 \leq 10^{-10}$とした．初期近似解$\boldsymbol{x}_0$はすべて0とした．行列は予め対角スケーリングによって対角項をすべて1.0に正規化した．最大反復回数は10000回とした．調べた反復法はGPBiCG法，AS_GPBiCG_v1法，AS_GPBiCG_v2法，GPBiCG_AR法，GPBiCGSafe法，BiCGSafe法の計6種類とし，前処理としてフィルインを考慮しないILU(0)分解を選んだ．初期シャドウ残差$\boldsymbol{r}_0^*$には初期残差$\boldsymbol{r}_0$を代入した．

3.2 実験結果

表**1-2**に5種類のILU(0)前処理つき積型反復法の収束性を示す．表中の"max"は最大反復回数までで収束しなかったことを意味し，"TRR"は真の相対残差(True Relative Residual)の常用対数$\log_{10}(||\boldsymbol{b} - A\boldsymbol{x}_{k+1}||_2/||\boldsymbol{b} - A\boldsymbol{x}_0||_2)$の値を意味する．最大反復回数に達するまでに収束しかつ近似解に対する真の相対残差の2ノルムの大きさTRRが$10^{-9.60}$以上の場合は偽収束(表中のTRRの数字に括弧をつけた)と判定した．表**3**に5種類の解法における収束(未収束)ケース，偽収束ケースと最速ケースを示す．表**4**に6種類の反復解法の速度順位とその総合順位を示す．ただし，成績表の集計について，計算時間が同じ場合は同

一順位にした．また収束しなかった場合は順位を6位にした．表の観察から以下の知見が得られる．

- GPBiCG_AR 法と GPBiCGSafe 法，BiCGSafe 法は全ての行列で収束したが，残り3種類の解法は反復回数が最大回数に達したり，収束しても偽収束現象が発生したときがあった．
- AS_GPBiCG_v1 法では行列 bcircuit，sme3Dc の2ケース，AS_GPBiCG_v2 法では行列 sme3Db の1ケースで偽収束が発生した．また GPBiCG 法で行列 sme3Db の1ケースで偽収束が発生した．
- 17種類のテスト行列において，最も速く収束したケースが多い解法は GPBiCGSafe 法の8ケースである．
- GPBiCG 法の最も速く収束したケースが零に対し，AS_GPBiCG_v1，_2 法は1ケースと2ケースであるが，GPBiCG 法よりも計算時間が長いケースが多い．
- 行列 bcircuit において，GPBiCGSafe 法の計算時間は GPBiCG_AR 法の計算時間より約 10%ほど速い．

Table 1: 6種類の ILU(0) 前処理つき積型反復法の収束性

行列	解法	反復回数	合計時間 [s]	平均反復時間 [ms]	TRR
air-cfl5	GPBiCG	21	7.032	76.619	-10.11
	AS_GPBiCG_v1	21	7.205	76.952	-10.11
	AS_GPBiCG_v2	21	7.190	77.429	-10.11
	GPBiCG_AR	21	6.776	76.714	-10.06
	GPBiCGSafe	21	**6.749**	77.190	-10.06
	BiCGSafe	21	6.787	76.762	-10.06
atmosmodd	GPBiCG	89	14.157	5.112	-10.08
	AS_GPBiCG_v1	90	14.821	5.011	-10.03
	AS_GPBiCG_v2	90	14.716	4.978	-10.10
	GPBiCGAR	90	13.341	5.156	-10.09
	GPBiCGSafe	90	**13.247**	5.067	-10.24
	BiCGSafe	93	13.685	4.935	-10.15
poisson3Db	GPBiCG	86	3.481	6.605	-10.12
	AS_GPBiCG_v1	84	3.440	6.714	-10.07
	AS_GPBiCG_v2	84	3.424	6.714	-10.08
	GPBiCGAR	84	3.383	6.738	-10.06
	GPBiCGSafe	84	3.369	6.714	-10.23
	BiCGSafe	83	**3.354**	6.916	-10.05

4 まとめ

本論文では，積型反復法で降順な漸化式で定義される多項式 $\{G_n(A)\}$ の存在に着目し，GPBiCG_AR 法において降順な漸化式を使用しない GPBiCGSafe 法を提案した．数値実験の結果，提案法が収束性において優れていることがわかった．

Table 2: 6 種類の ILU(0) 前処理つき積型反復法の収束性 (cont'd)

行列	解法	反復回数	合計時間 [s]	平均反復時間 [ms]	TRR
raefsky3	GPBiCG	129	1.759	1.589	-10.04
	AS_GPBiCG_v1	133	1.803	1.534	-10.80
	AS_GPBiCG_v2	125	**1.711**	1.656	-10.23
	GPBiCG_AR	134	1.803	1.507	-10.65
	GPBiCGSafe	134	1.806	1.537	-10.36
	BiCGSafe	140	1.882	1.479	-10.28
water_tank	GPBiCG	270	5.260	0.841	-10.02
	AS_GPBiCG_v1	285	5.537	0.800	-10.27
	AS_GPBiCG_v2	276	5.344	0.812	-10.02
	GPBiCG_AR	279	5.338	0.781	-10.29
	GPBiCGSafe	269	**5.154**	0.855	-10.01
	BiCGSafe	275	5.260	0.822	-10.29
bcircuit	GPBiCG	6995	55.321	0.004	-10.01
	AS_GPBiCG_v1	4765	38.645	0.007	(-9.41)
	AS_GPBiCG_v2	4553	36.275	0.007	-10.18
	GPBiCG_AR	3594	27.715	0.010	-10.52
	GPBiCGSafe	3239	**24.851**	0.009	-10.14
	BiCGSafe	4114	32.091	0.008	-10.21
Freescale1	GPBiCG	1681	675.306	1.071	-10.09
	AS_GPBiCG_v1	1567	659.221	1.154	-10.05
	AS_GPBiCG_v2	2135	890.542	0.834	-9.94
	GPBiCG_AR	1347	**500.602**	1.325	-10.02
	GPBiCGSafe	1384	512.137	1.293	-10.11
	BiCGSafe	1409	531.936	1.276	-10.11
memplus	GPBiCG	251	0.390	0.044	-10.04
	AS_GPBiCG_v1	257	0.416	0.047	-10.04
	AS_GPBiCG_v2	243	0.398	0.049	-10.03
	GPBiCG_AR	251	0.379	0.052	-10.08
	GPBiCGSafe	244	**0.363**	0.045	-10.13
	BiCGSafe	248	0.377	0.044	-10.09

Table 3: 6 種類の解法における収束 (未収束) ケース，偽収束と最速ケース

解法	収束ケース	未収束のケース	偽収束ケース	最速のケース
GPBiCG	15/17	2/17	1	0/17
AS_GPBiCG_v1	15	2	2	0
AS_GPBiCG_v2	14	3	1	2
GPBiCG_AR	**17**	0	0	4
GPBiCGSafe	**17**	0	0	**8**
BiCGSafe	**17**	0	0	3

Table 4: 6種類の反復解法の速度順位とその総合順位

解法	順位						合計	総合順位
	1	2	3	4	5	6		
GPBiCG	0	2	3	3	6	3	73	4
AS_GPBiCG_v1	0	1	1	8	1	6	78	5
AS_GPBiCG_v2	2	0	0	4	4	7	80	6
GPBiCG_AR	4	5	5	1	2	0	43	2
GPBiCGSafe	8	4	3	0	2	0	**35**	**1**
BiCGSafe	3	6	5	1	0	2	46	3
total	17	18	17	17	15	18	-	-

References

[1] S.-L. Zhang, GPBi-CG: Generalized product-type methods preconditionings based on Bi-CG for solving nonsymmetric linear systems, SIAM J. Sci. Comput., pp.537-551, 1997.

[2] 藤野 清次, 藤原 牧, 吉田 正浩, 準残差の最小化に基づく BiCGSafe 法の収束性について,Transactions of JSCES, Vol.2005, 20050028, 2005.

[3] Moethuthu, S. Fujino: Stability of GPBiCG_AR method based on minimization of associate residual, Journal of ASCM, pp.108-120, 2008.

[4] K. Abe, G.L.G. Sleijpen, Solving linear equations with a stabilized GPBiCG method, 日本応用数理学会 第 14 回環瀬戸内応用数理研究部会, 岡山理科大学, Jan., 2011.